UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/650,238	08/27/2003	Wolfram Schulte	3382-65592	6392

26119 7590 08/28/2007

KLARQUIST SPARKMAN LLP
121 S.W. SALMON STREET
SUITE 1600
PORTLAND, OR 97204

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
08/28/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/650,238	SCHULTE ET AL.	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *19 June 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-33* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-33* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *7/31/07*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 6/19/07.

As indicated in Applicant's response, claims 1, 18-19, 33 have been amended. Claims 1-33 are pending in the office action.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 1-33 are rejected under 35 U.S.C. 102(b) as being anticipated by Davidson et al., USPN: 6,083,276(hereinafter Davidson).

As per claim 1, Davidson discloses a computer implemented method for producing a data domain for a data structure element of a computer program, the method comprising:

receiving domain configuration information corresponding to the data structure element (e.g. Fig. 3A, 3B, 3C);

receiving a reflection of the computer program (e.g. bean objects 212- Fig. 2); and

producing the data domain (e.g. Fig. 4B, 4C – Note: mapping corresponding descriptor or attribute for a method or class reads on data domain combining configuration information with reflection of beans components – see col. 25, line 12 to col 26, line 9) based on the domain configuration information and the program reflection, and

targeting testing of the computer program to use only values for the data structure element that fall within the data structure (e.g. *expected parameters* – col. 25, lines 8-32;

canonical names ... scope path named ... more attributes ... remain to be mapped – col. 25, line 34 to col. 26, line 9 – Note: verifying correctness of parameters, basic types, canonical objects in light of their expected number or instances being enclosed within some Descriptor scope reads on testing computer program so that data structure domain values fall under that structure – see propertyDescriptor, parameters … write method - Fig. 4C).

As per claim 2, see Fig. 1 for computer readable media having computer executable instructions for performing the method of claim 1.

As per claim 3, refer to claim 1 for a listing of data structure elements of the computer program as reflection of computer program (e.g. Fig. 2).

As per claim 4, Davidson discloses annotating code of the computer program (e.g. comment 302 – Figs. 3; e.g. col. 29-36 Appendix A for ADML for comments between special tags <!-- ... -->) with the domain configuration information.

As per claims 5-6, Davidson discloses computer readable media having computer executable instructions for compiling the code of the computer program annotated with the domain configuration information for producing the data domain (Fig. 4B, 4C cols 21-28) according to its domain configuration information.

As per claims 7-8, Davidson discloses the domain configuration information comprising one or more expressions (e.g. BML – col. 8-col. 10 – Note: tag specification <*Foo Att1 = Value 1 Att2=Value2 ...* /> reads on explicit denotation of domain to be produced) for explicitly denoting the data domain to be produced corresponding in form to one that is applicable to the data structure element; wherein the expressions comprise methods and functions (e.g. parameter method Fig. 4C; METHOD, ARGUMENTS - col. 16, lines 28-42; Fig. 5; *CALL calls a method*

... *Attributes* - Appendix A, col. 47, bottom - Note: beans constructs being described in BML language as method and arguments reads on methods and functions) defined within the code of the computer program, which are exposed via the reflection of the computer program.

As per claim 9, Davidson discloses wherein the data structure element is a data type with one or more fields and the form of the explicitly expressed data domain is a set of values of the fields comprising the data type (e.g. TYPE – col. 16, lines 54-63).

As per claim 10, Davidson discloses wherein the data structure element is a method (re claim 9) and the form of the explicitly expressed data domain is a set of tuples of parameters (e.g. *GET FIND CONSTANT|ARRAY| ANY| ALL|NOT* – Appendix A, col. 29-30) of the method.

As per claim 11, Davidson discloses wherein the data structure element is a field or a parameter of a designated type (Table 2, pg. 19; TYPE ID - lines 55-63, col. 16) and the form of the explicitly expressed data domain is an enumeration of values (e.g. <VALUE ...</VALUE> lines 55-63, col. 16; lines 28-42, col. 16) of the designated type corresponding to the field or the parameter.

As per claim 12, Davidson discloses inheriting (e.g. Fig. 3C; *children* - col. 10, lines 50-65; *Child component ...Parent component* – Fig. 4D; lines 9-29 - col. 13) the data domain to be produced from the data domain of other related data structure elements.

As per claim 13, Davidson discloses a data type comprising a plurality of sub-types and a selection of one or more of the plurality of sub-types wherein the data domain to be produced for the data type is a union of data domains of the sub-types (P tag ... Table 2, col. 19; Style tag...Table 3, col. 20 – Note: paragraph and style tag with subtypes read on plurality of subtype and selection from an union of subtypes) belonging to the selection.

As per claim 14, Davidson discloses data structure element being a field or a parameter of a designated type (Fig. 5; Table 2, pg. 19; TYPE ID - lines 55-63, col. 16; cols. 17-18) and the domain configuration information comprises information indicating that the data domain to be produced for the field or the parameter is inherited (e.g. Fig. 3C; *children* - col. 10, lines 50-65; *Child component... Parent component* – Fig. 4D; lines. 9-29 - col. 13)from the data domain of their designated type.

As per claims 15-16, Davidson discloses domain configuration information related to producing the data domain for the data structure element by applying domain generation techniques on other selected data domains; and filtering the result of the applying domain generation technique step using a predicate (steps 410, 418, Fig. 4B; Match 424, Conversion 432, More attributes 440 – Fig. 4C; step 456, Fig. 4D – Note: code to map components as called by description information with respect to parse algorithm reads on predicates for filtering data from information domain into data domain – see Java pseudo-code col. 26, 28).

As per claims 17-18, Davidson discloses data structure element is a data type with a plurality of fields (lines 30-34, 55-58 -col. 9; lines 30-43, col. 16; Table 2, col. 19) and the other data domains are data domains of the fields (re claim 1 or Fig. 4B, 4C); wherein the data structure element is a method (Table 1, col. 17-18) and the other data domains are data domains of the parameter of the method (re claim 1; see *write method 504* - col. 25).

As per claim 19, Davidson discloses a system for producing a data domain for a data structure element of a computer program, the system comprising a computer apparatus configured to perform actions of a domain configuration manager for

receiving domain configuration information (e.g. 3A, 3B, 3C) corresponding to the data structure element and

using a reflection of the computer program (e.g. bean objects 212- Fig. 2) to produce the data domain for the data structure element according to the domain configuration information (Fig. 4B, 4C – Note: mapping corresponding descriptor or attribute for a method or class reads on data domain combining configuration information with reflection of beans components); and

controlling testing of the computer program to use only values for the data structure element that fall within the data structure (e.g. *expected parameters* – col. 25, lines 8-32; *canonical names ... scope path named ... more attributes ... remain to be mapped* – col. 25, line 34 to col. 26, line 9 – Note: verifying correctness of parameters, basic types, canonical objects in light of their expected number or instances being enclosed within some Descriptor scope reads on testing computer program so that data structure domain values fall under that structure – see propertyDescriptor, parameters ... write method - Fig. 4C).

As per claims 20-21, Davidson discloses a graphical user interface communicative with the domain configuration manager for receiving the domain configuration information and transferring (Fig. 1; col. 18-40 -col.28; Error Message – Fig 4B) the domain configuration information to the domain configuration manager; a GUI for receiving user input related to the domain configuration information (e.g. user and application-generated 'events' lines 15-40 -col. 10).

As per claim 22, Davidson discloses domain configuration manager for reading the reflection of the computer program to identify the data structure element for its domain configuration (e.g. Fig. 3B, 3C; Fig. 4B).

As per claim 23, Davidson discloses wherein the data structure element is a data type and the domain configuration manager is operable for producing the data domain for the data type according to an explicit expression indicative of the data domain of the data type (refer to rationale of claims 13-14).

As per claim 24, Davidson discloses wherein the explicit expression comprises methods and functions defined within the computer program (e.g. col. 17-18) and exposed to the domain configuration manager via the reflection of the computer program (Table 1, col. 17-18).

As per claim 25 Davidson discloses wherein the data structure element is a method and the domain configuration manager is operable for producing the data domain as a set of tuples of parameters of the method according to an explicit expression of the domain configuration information (refer to claim 10).

As per claim 26 Davidson discloses wherein the data structure element is a field or a parameter of a declared type and the data configuration manager is operable for producing the data domain according to an explicit expression whose result is an enumeration of values of the declared type (refer to claim 11).

As per claim 27 Davidson discloses wherein the data structure element is a data type with sub-types and the data configuration manager is operable for producing the data domain for the data type through inheritance as a union of data domains of its selected sub-types (refer to claim 13).

As per claim 28 Davidson discloses wherein the data structure element is a data type and the data configuration manager is operable for producing the data domain for the data type by

applying a domain generation technique to one or more fields of the data type (refer to claim 15).

As per claim 29, Davidson discloses wherein the domain generation technique is a Cartesian product (e.g. *mapper 122, 124* – Fig. 1; *map 418* – Fig. 4b; *Mach 424*, Fig. 4C; *BeanInfo Mapper* - Fig. 5 - Note: a Cartesian or Cross product between 2 sets A and B is defined as the set of all pairs $\{a, b\}$ such that a is an element of the set A and b is an element of the set B; i.e. mapping an element of A with a corresponding element of B) of the selected fields of the data type and the domain configuration manager is further operable for applying a constraint specified in the domain configuration information to the Cartesian product for producing the data domain for the data type (refer to claim 16 for filtering constraint using predicate).

As per claim 30 Davidson discloses wherein the data structure element is a field or a parameter of a declared type and the domain configuration manager is operable for producing the data domain for the field or the parameter as the data domain of their respective declared type through inheritance (refer to claim 14).

As per claim 31, Davidson discloses wherein the data structure element is a method (re claim 24-25) and the domain configuration manager is operable for producing the data domain for the method by applying a domain generation technique to the parameters of the method (re claim 15).

As per claim 32, Davidson discloses wherein the domain configuration technique is a Cartesian product (re claim 29) of the data domains of the parameters (re claims 26 and 30) of the method and the data configuration manager is further operable for applying a constraint (refer

to claim 16) for filtering constraint using predicate) to the result of the Cartesian product for producing the data domain for the data type.

As per claim 33, Davidson discloses a computer-based system for producing data domains of data structure elements of a computer program, the system comprising a computer apparatus; and

means for:

receiving, on the computer apparatus, domain configuration information corresponding to the data structure elements;

reading, on the computer apparatus, a reflection of the computer program; and

processing, on the computer apparatus, the domain configuration information and the reflection to produce and output the data domains (e.g. Fig. 1, 3; Fig .5 – Note: attributes and their expected instances within Descriptor scope reads on data domain being **outputted** for verification via mapping) corresponding to the data structure elements; and

for limiting testing of the computer program to use only values for the data structure element that fall within the data structure;

all of which limitations having been addressed in claim 1.

Response to Arguments

4. Applicant's arguments filed 6/19/07 have been fully considered but they are not persuasive. Following are the Examiner's observation in regard thereto.

35 USC § 102 Rejection:

5. Applicants have submitted that for claim 1, Davidson's creating and configuring of an application through a text based grammar does not produce a 'data domain for a data structure

element of a computer program' based on 'domain configuration' and 'program reflection' (Appl. Rmrks pg. 10, middle). There is nothing particular teaching in the claim language that enforces a more narrower nature to this 'data domain' in such terms that would render inapposite the text-based attribute discovery and mapping in Davidson's program configuration and verification. The phrase recited as 'data domain for a data structure element of a computer program' at best entails existence of a data structure related to a software program for which there is a data set having some specific domain type of connotation or grammar construction scope (e.g a type variance, a parameter range), and while the term 'domain' is not described with further distinguishing properties, its interpretation remains that of a set of insight or data knowledge of a particular nature; and Davidson's set of attributes being discovered from parsing a Descriptor does fulfill on the connotation of a scope, a range of a type of attributes or canonical objects. The rejection has pointed to portions by which based of a specification of a Descriptor object, the program or method parameters, attributes (see Fig. 4C, col. 25-26) are checked to see if they are exhausted from their expected scope, or intended definite set called by the Descriptor, all of which connotating a range or a domain for their being beyond which the mapping tool would not consider these parameters/attributes valid or syntactically/rule compliant. That is, the *attribute mapper* by Davidson has applied verification of program constructs based on some 'descriptor' structure, and such testing is retrieving and utilizing the canonical representation of the properties or attributes based on what is expected from them (via parsing of tree), the basis of this expectation or compliance being founded from interpreting a Descriptor (via a parsing process of the tool) being a data structure element. And in validating these canonical form or attributes/names, the mapper ensures that all of these are checked and mapped against what is

required from the Descriptor. For one of ordinary skill in the art faced with the term 'domain data', the concept of a *domain* can be construed in the fact that the expected instances of these canonical form of attributes names, or parameters amount to a numerical or contextual scope thereof, e.g. a range of instances of attributes called for by the Descriptor and generated during the runtime of the mapper. It is deemed that the claim cannot preclude the expected range of attributes being fetched by the mapper for compliancy purpose as cited in the rejection to fulfill what 'data domain' is all about, absent specificity in the claim about what exactly 'domain' constitutes of; thus, it is deemed that Davidson has met the 'producing of data domain ... based on domain configuration configuration and program reflexion' as set forth in the rejection. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

(B) Applicants have submitted that there has no indication that Davidson' s descriptor or attribute contains the word 'domain' from which to derive *Descriptor* and *attribute* (Appl. Rmrks pg. 11, bottom). In reply, the term domain is not recited with more in-depth teaching to enable a more narrow construction as to its very nature beyond the superficial layer of its textual name or hollow nomenclature. Broad interpretation has it that 'data domain' is to be viewed as a set of insightful instances or data knowledge of a particular nature, with a set of some range beyond which the set would no more have its meaning. Thus, Davidson's attributes when called for a specific program construct, entail a set thereof (all the attributes required for a method call, no more nor less) for which the mapper would consider compliant; and this set reads on domain. Broad and reasonable interpretation has been used to interpret a broadly claimed term; the

concept of *domain* has been taught in the way Davidson's attributes are organized to represent a bounded domain/scope restricted by the rule of the Java inference engine or language grammar (e.g. object-oriented inheritance or object/class constructor); therefore, the above argument is not persuasive.

(C) Applicants have submitted that Davidson's Figure 4 are directed to generate an application, not data domain, nor are they for receiving a reflexion and produce a data domain based thereon, because there is no computer program at the point in Davidson's techniques at which data domains are alleged to be produced (Appl. Rmrks pg. 11, middle). The argument seems contradictory in that Davidson's tool is viewed as to merely generating a Java application program and at the same time not having/generating a computer program for which to create data domains, thus not in a readable form for a proper reply. As far as Davidson not generating a data domain, this argument is to be referred back to sections A and B. Unless the claim language specifies what a *data domain* (related to program structure element) constitutes of, one of ordinary skill in the software arts would perceive program structure element as one that is very broad in meaning, and any 'data domain' related to this program structure element can be as broad as the range of a (Java) base type, compound type, or a family child objects expected from a parent class; because, *inter alia*, a type or a child object can encompass a certain limit beyond which the type or the object instance might cease to qualify itself as being itself with respect to the software language rule (e.g. type integer, type complex, private class type, abstract type class). Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

(D) Applicants have submitted that claims 19 and 33 are allowable in light of Applicants' reasons discussed above regarding the deficiency of Davidson (Appl. Rmrks pg. 11-12). In reply, the argument is referred back to the above 3 sections. Davidson is deemed to have met claims 19 and 33 as a consequence of the above observations.

The claims will stand rejected as set forth in the Office Action.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (571) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571)272-3756.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Tuan A Vu
Patent Examiner,
Art Unit 2193
August 21, 2007